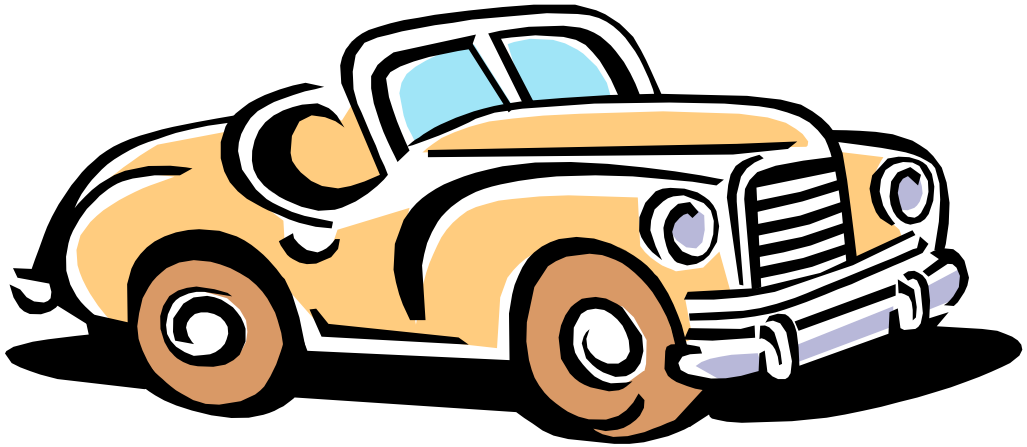
# Maximum RPSTL

Getting the Most Out of the
Latest RPSTL Releases

# ‘Current’ RPSTL

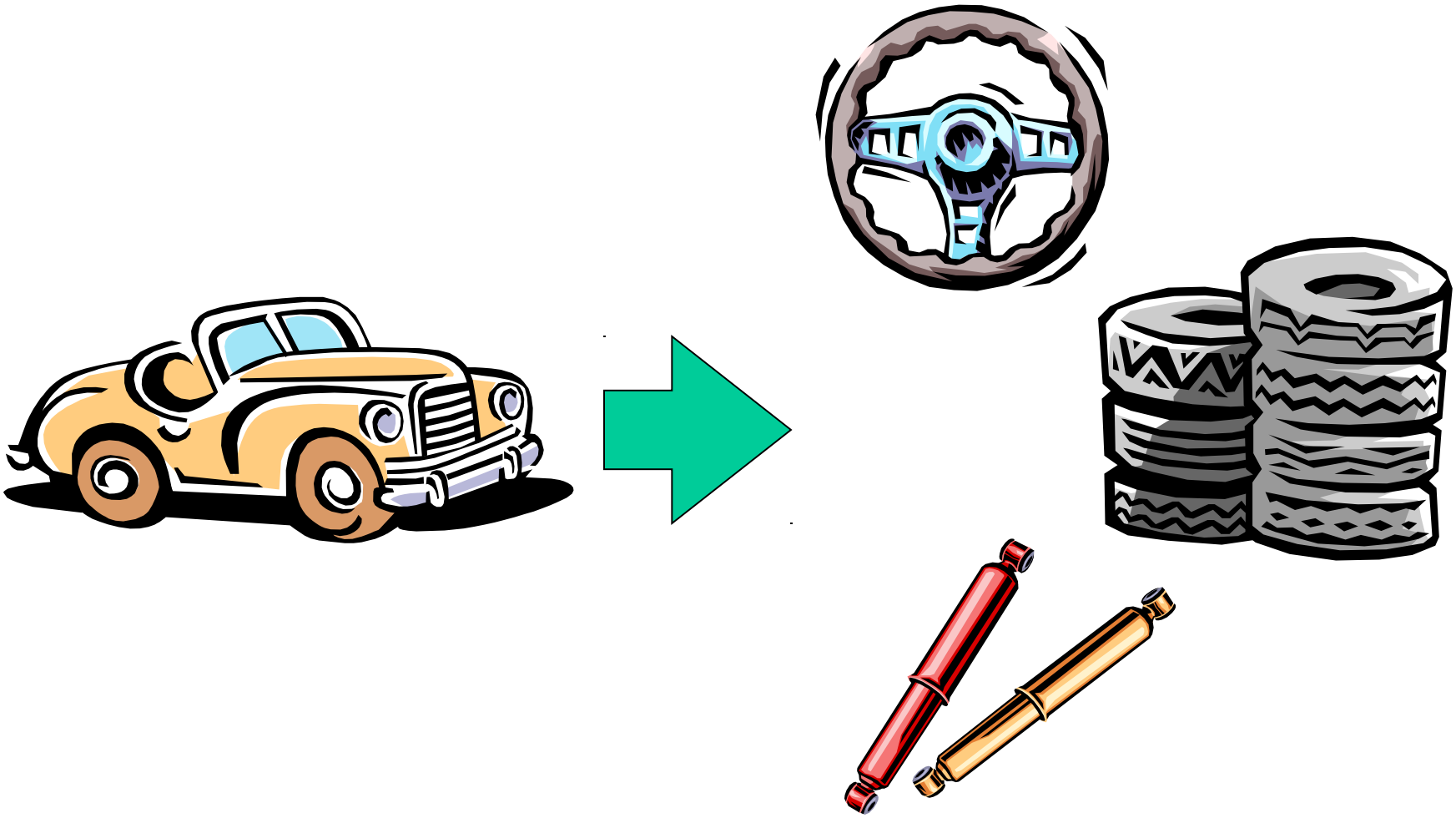


‘Current’ RPSTL is assumed to be V3.0

# Agenda/Major Topics

- User interface upgrades in Author and Reader
- Customizing the look of a RPSTL
- Using custom interfaces in RPSTL
- Looking ahead: Ongoing evolution of RPSTL
- Question and answer session

# Interface Improvements

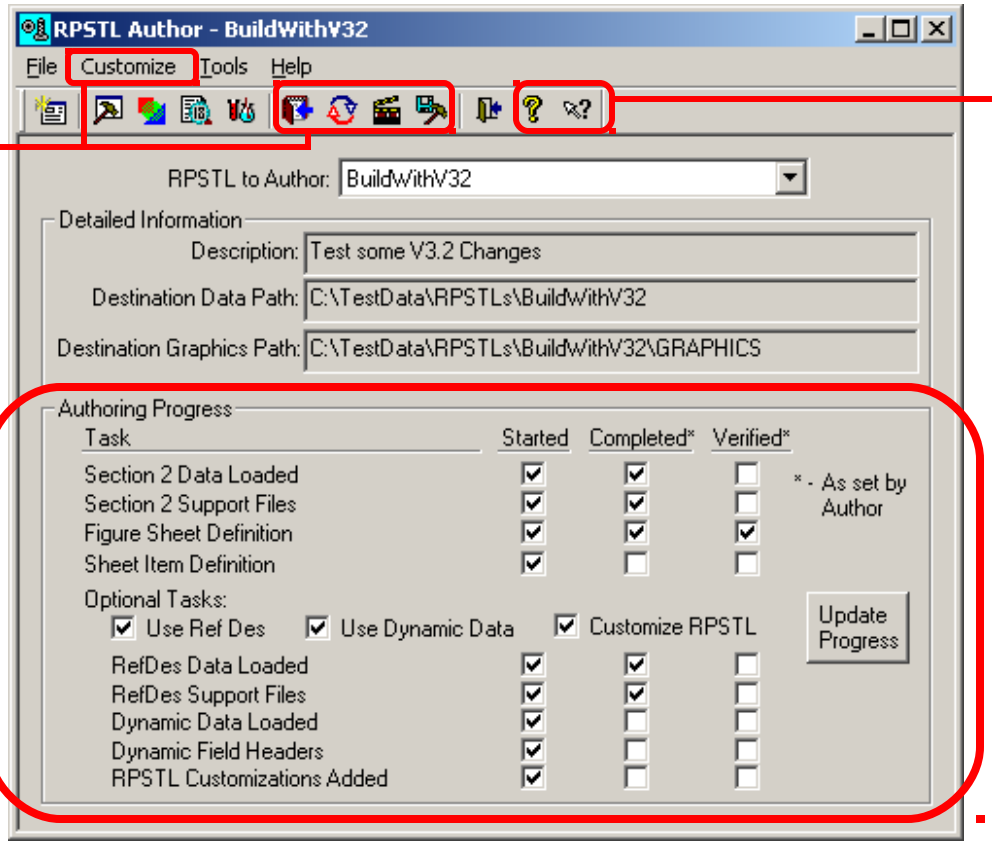


# New Interface Features

- Authoring Interface Features
  - Separate Main Window with Authoring Progress
  - Customize Menu and Buttons
- Reader Interface Features
  - Search – Figure/Item, Description, Sort by NIIN
  - Find All feature
  - Graphic Region Highlighting
- Miscellaneous Features
  - Accelerator Keys
  - Context Sensitive Help
  - Hotspot Utility
  - FigSheet Utility
  - CGM Utility (in ZoomView)

# RPSTL Author Features

New Customization Options

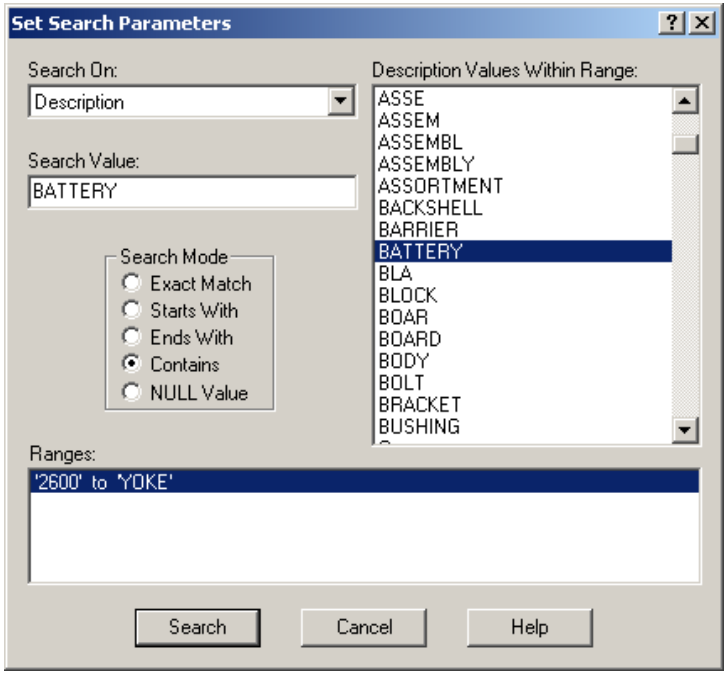



New Help Options

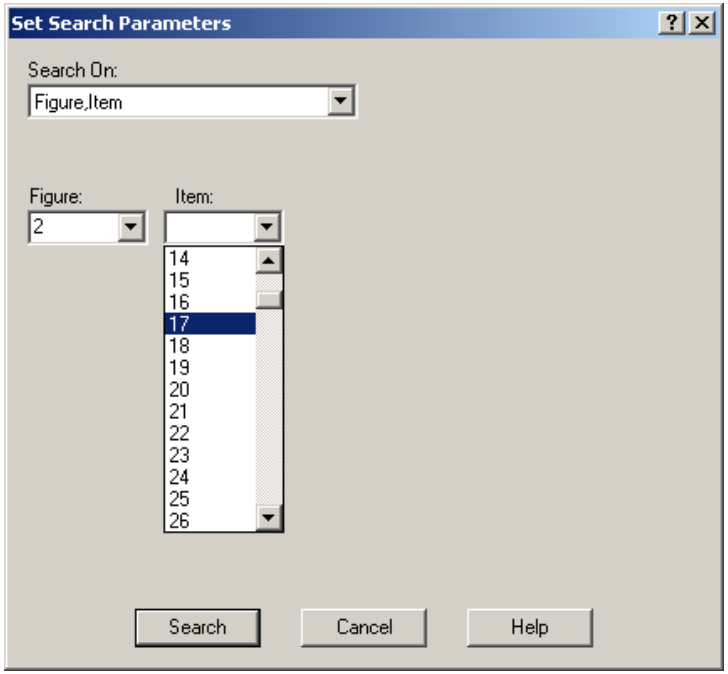
Authoring Progress Section

New Initial Dialog

# RPSTL Reader Features

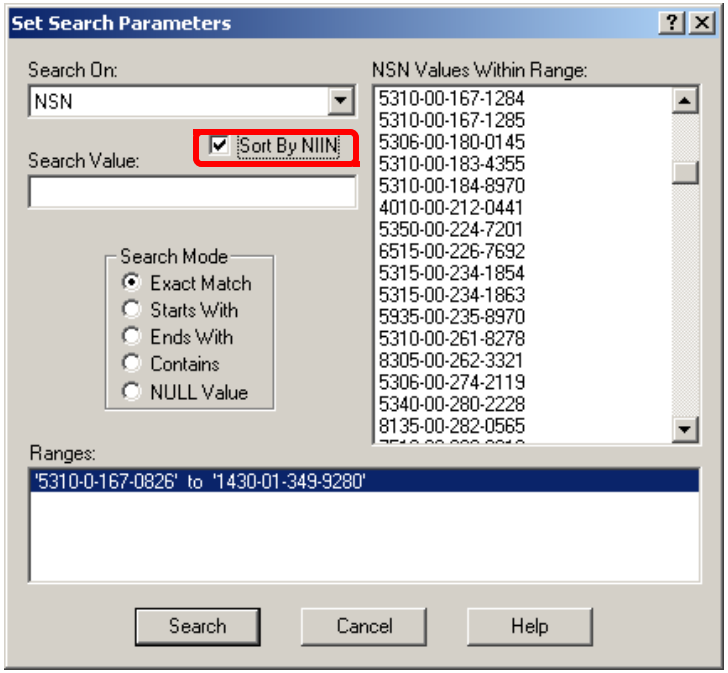



New Description Search

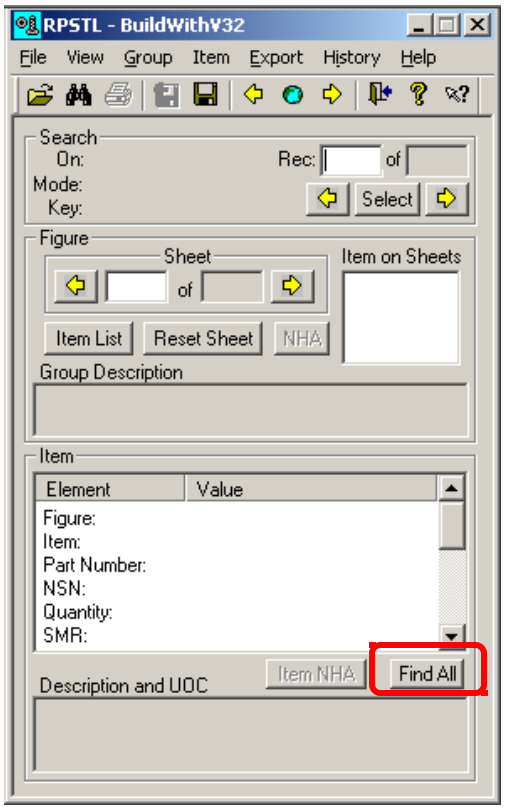



New Figure, Item Search

# Reader Features (continued)




Sort by NIIN Feature (NSN)

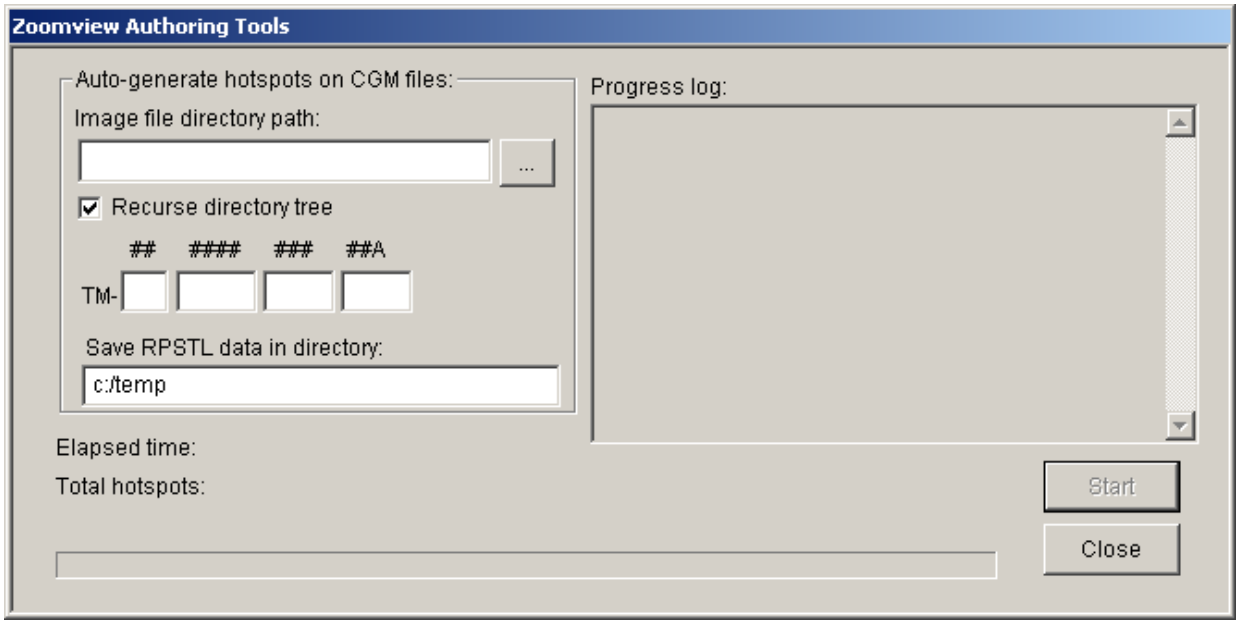



New Find All Button

# Utilities

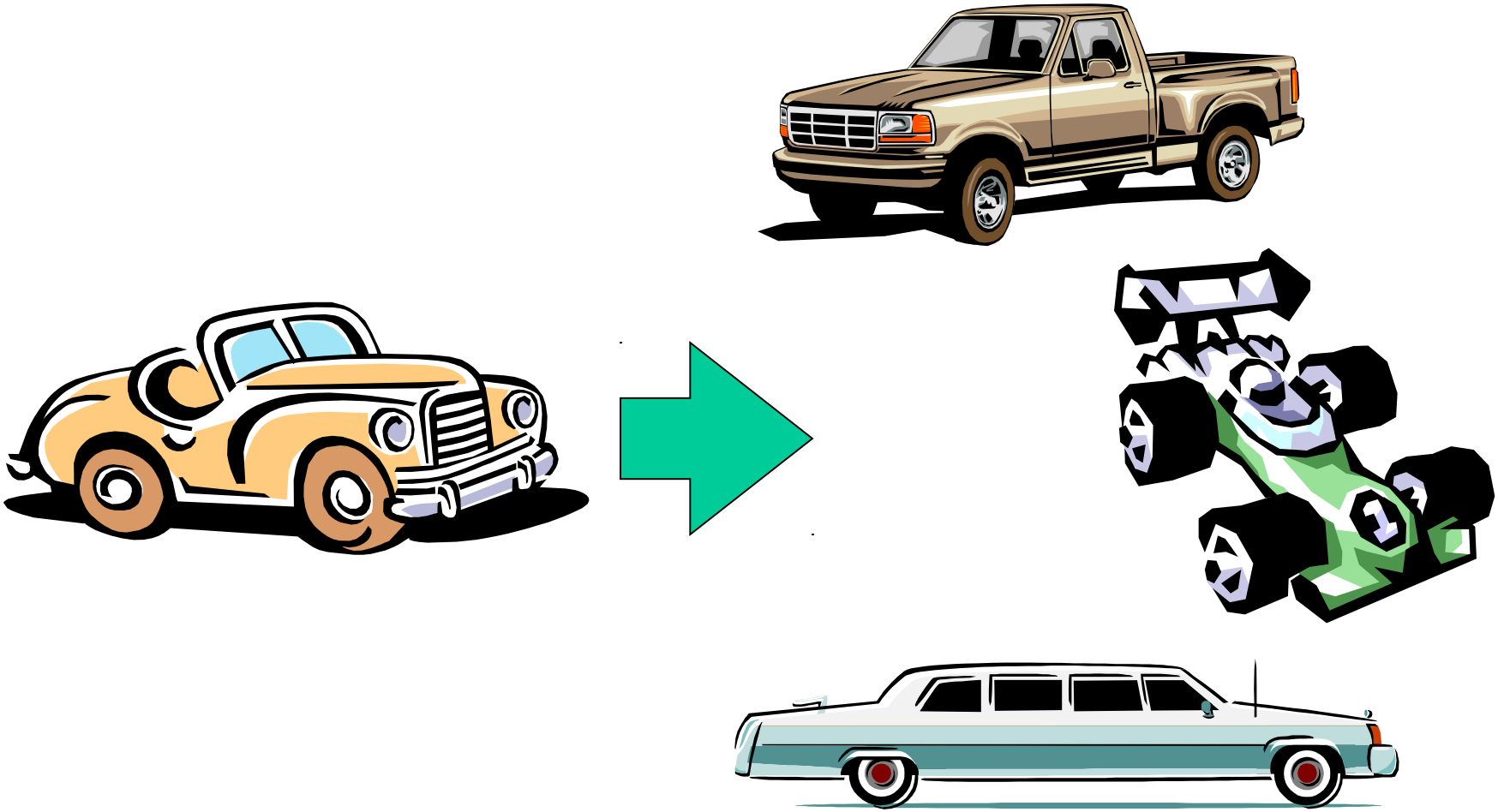
FigSheet Utility

RPSTL Hotspot Utility (RHU)




CGM Utility (under Tools menu in ZoomView Author)

# RPSTL Customization



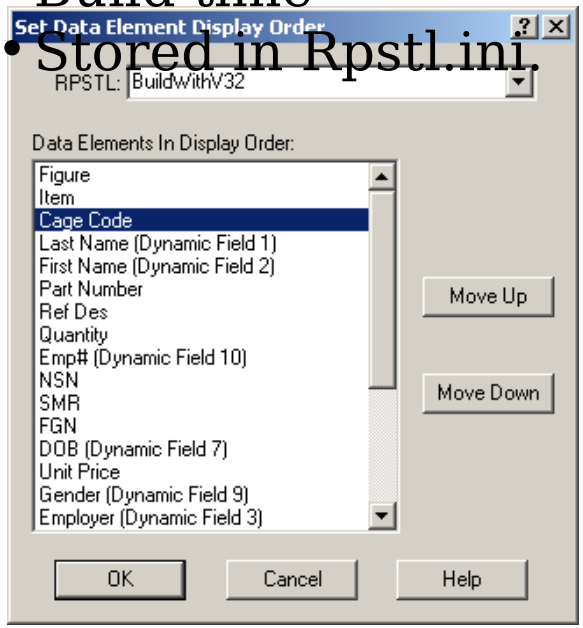
# Basic Customization Features

- Data Element Display – Setting element order and excluding unused elements.
- Front Matter – Setting a custom cover, table of contents, and RPSTL section 1.
- View Options – Setting the size and location of the RPSTL window.
- Next Higher Assembly – NHA definition at the Group and Item level.
- Break the Rules – Enable multiple figures per group.

# Data Element Display

**Excluding unused data elements**

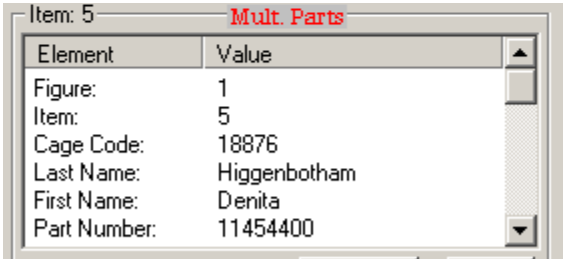
- Done automatically at Build time
- Stored in Rpstl.ini.



Button and dialog to set data element order.

**Set data element order**

- Includes standard elements and dynamic fields.
- Best performed after the Build step.
- Figure and Item are fixed as the first two data elements.
- Stored in Rpstl.ini

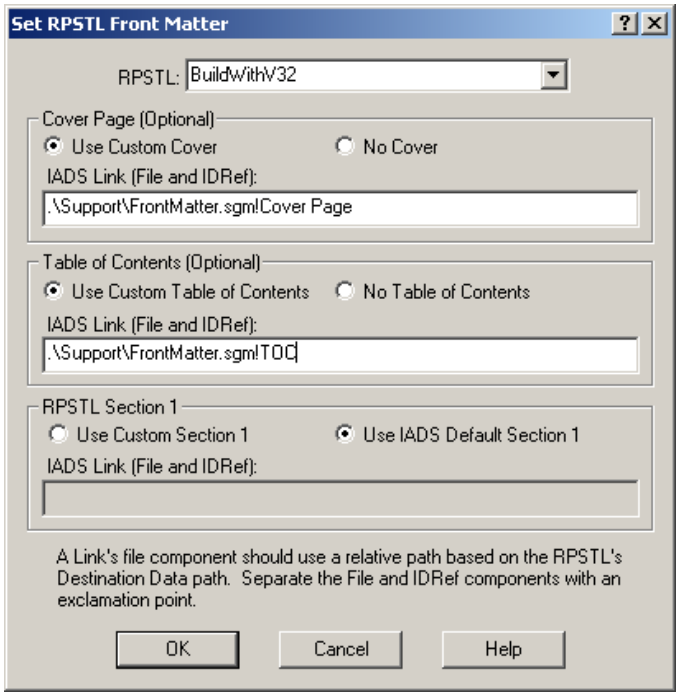



RPSTL Reader segment demonstrating the element order feature.

# Custom Front Matter

- Default Section 1 used if no custom document.
- Resolves relative paths using the destination data directory.
- Must reference an IADS file.
- Recognizes parameter values.

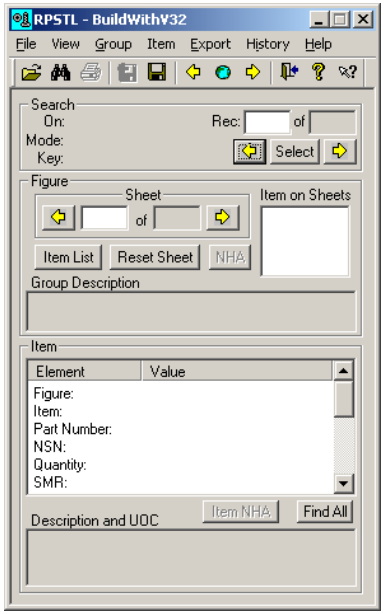
RPSTL Reader menu sequence to view one of the custom documents.

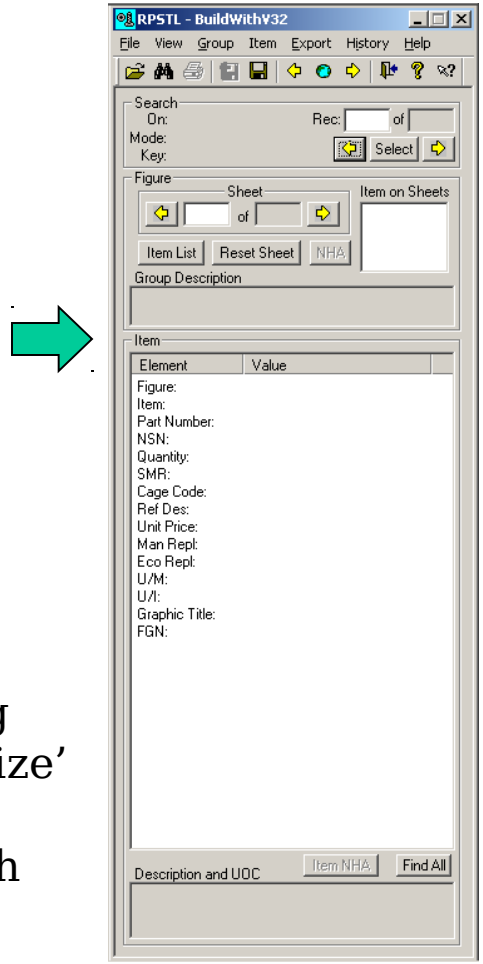

Set RPSTL Front Matter
RPSTL: BuildWithV32
Cover Page (Optional)
Use Custom Cover
No Cover
IADS Link (File and IDRef):
.\Support\FrontMatter.sgm!Cover Page
Table of Contents (Optional)
Use Custom Table of Contents
No Table of Contents
IADS Link (File and IDRef):
.\Support\FrontMatter.sgm!TOC
RPSTL Section 1
Use Custom Section 1
Use IADS Default Section 1
IADS Link (File and IDRef):
A Link's file component should use a relative path based on the RPSTL's Destination Data path. Separate the File and IDRef components with an exclamation point.
OK
Cancel
Help

RPSTL Author button and dialog to set any custom front matter.

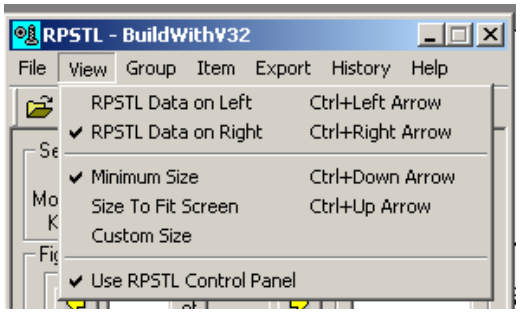
# View Options

- Specified by the RPSTL Reader user at run time.
- ‘Remembered’ by RPSTL Reader and restored next session.
- Selective resizing maximizes only critical regions.
- Enforces a minimum size.



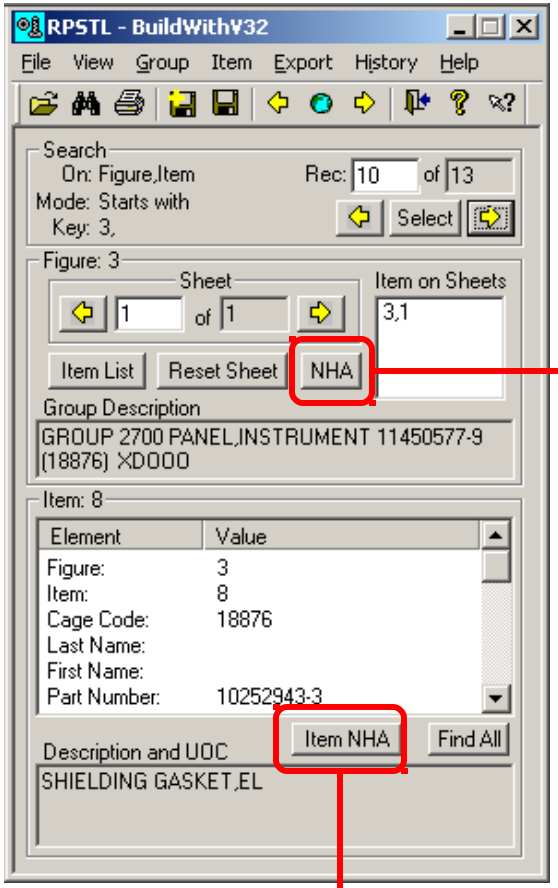

View Menu Options


RPSTL - BuildWithV32
File View Group Item Export History Help
RPSTL Data on Left Ctrl+Left Arrow
RPSTL Data on Right Ctrl+Right Arrow
Minimum Size Ctrl+Down Arrow
Size To Fit Screen Ctrl+Up Arrow
Custom Size
Use RPSTL Control Panel

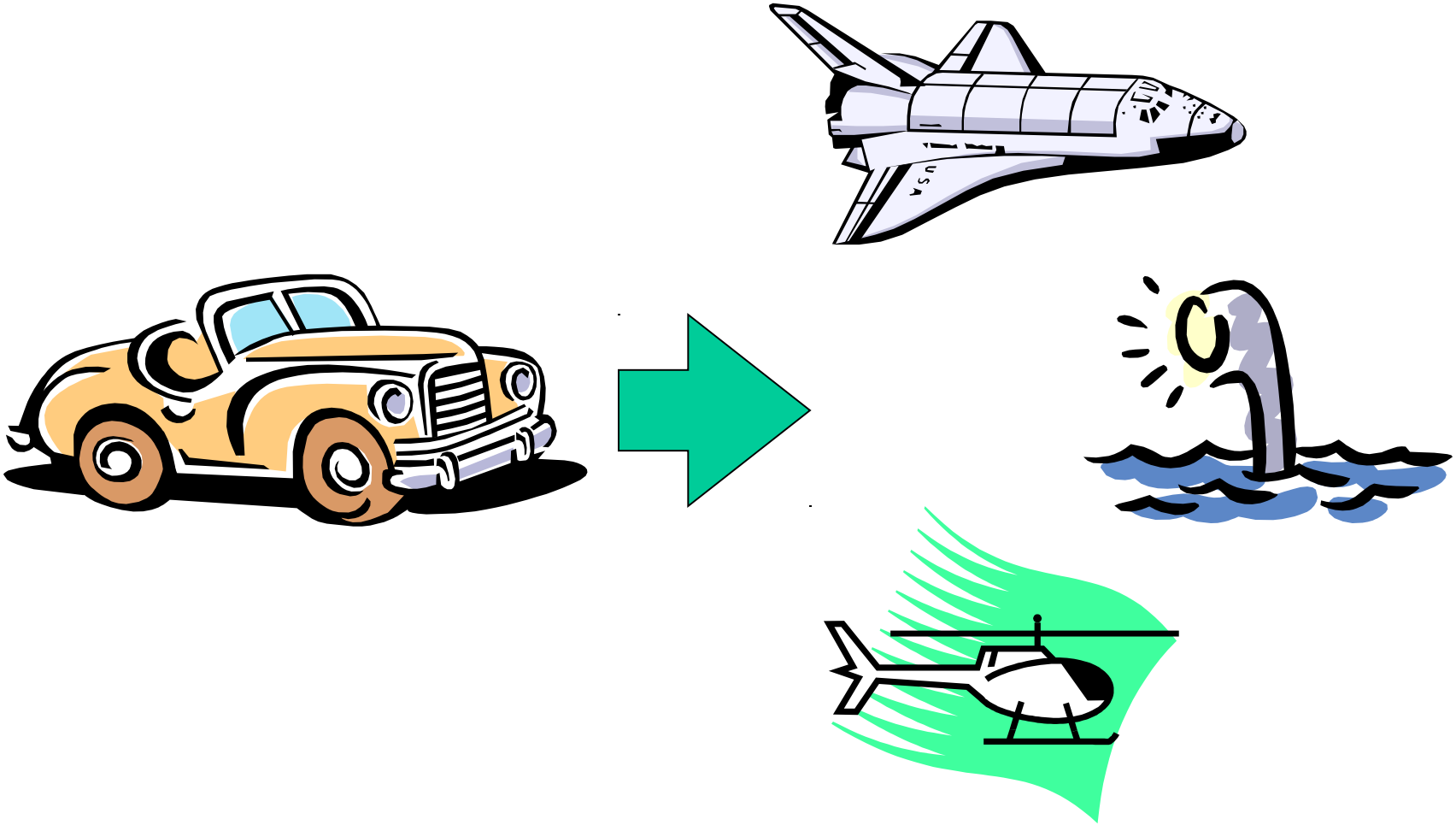
Effect of changing from ‘Minimum Size’ to ‘Size to Fit’ (which uses all available screen height).

# Next Higher Assembly

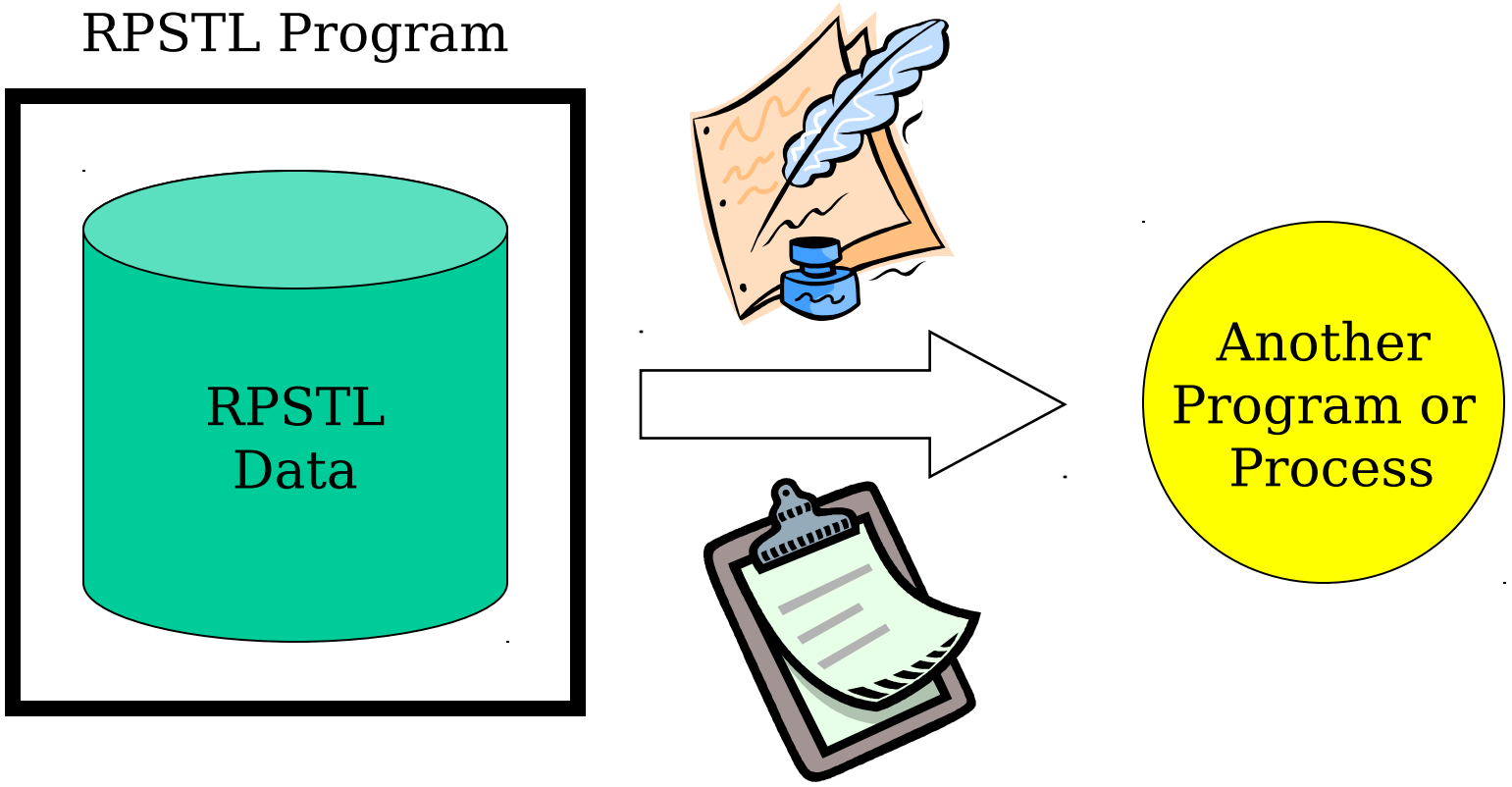
- Can be defined at Group and Item level.
- Established within the section 2 tag file and incorporated at Build time.
- Group level always specifies another group as the NHA.
- Item level uses many data elements including dynamic fields as the NHA (dictated by a processing instruction within the tag file).
- Buttons and menu options are disabled when a Group or Item NHA is not specified for an Item.



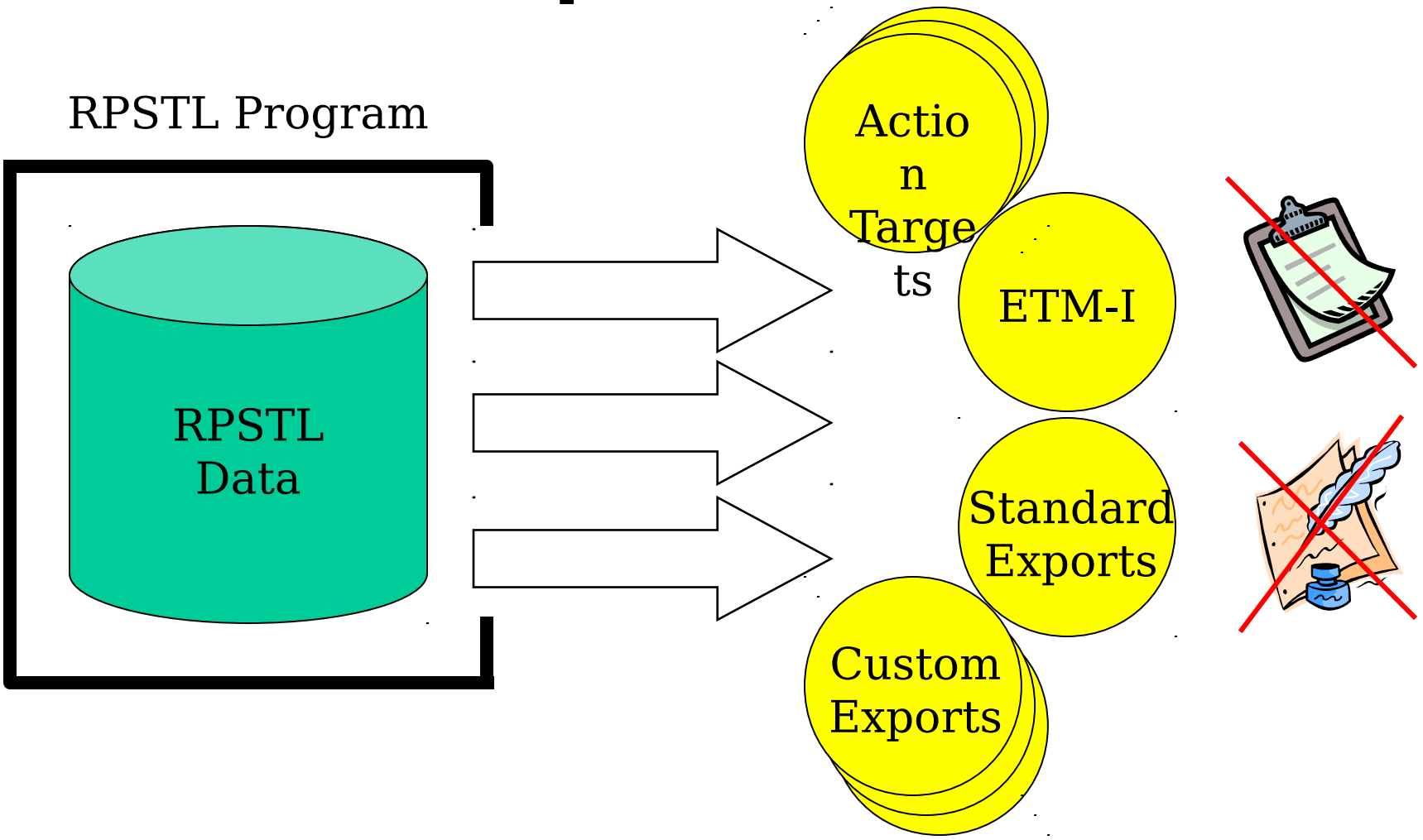
# Custom Interfaces



# Prior Interface Capabilities



# Using the New Capabilities



# Custom Interface Capabilities

- Internet Data Sources – Support for secure or password-based access for downloading Internet-based RPSTLs.
- Data Element Action – Pass a single data element value to an outside process.
- Export Function – Pass a range of part information for one or more parts to an outside process.
- Custom Export Types – Permit each RPSTL to establish customized export recipients.

# Compare and Contrast

Similarities between the Data Element Action and Export functions:

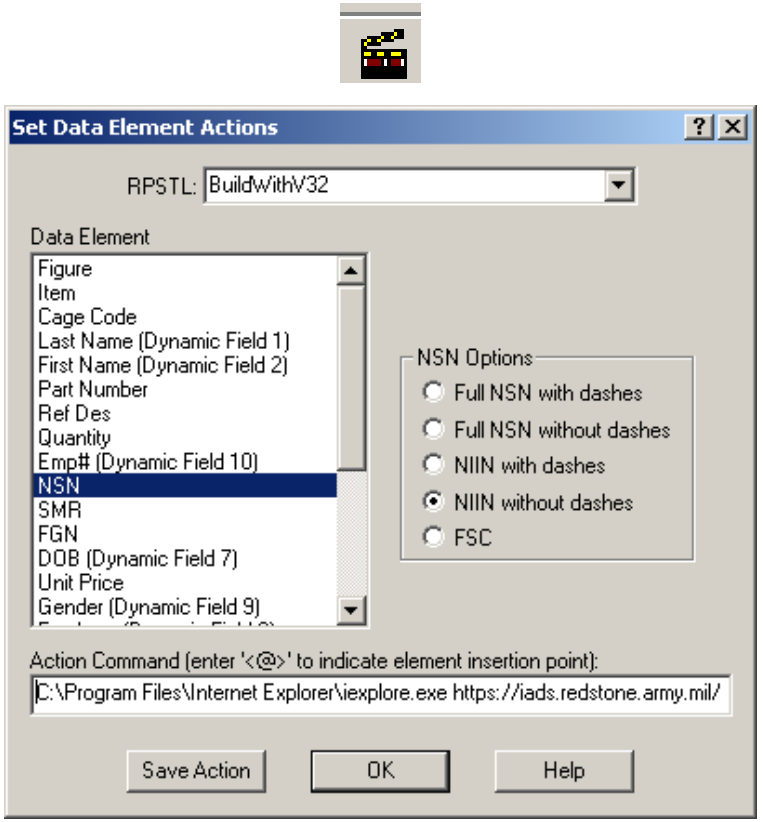
- Key mechanisms for sharing RPSTL data with outside processes.
- Accomplish data passing without user intervention.
- Custom exports and element actions require entry of a command to execute.
- Both commands should accept a passed data value.

Differences:

| Data Element Action | Export |
|---|---|
| Simple, quick to activate | Multiple steps to use |
| Single data element of a single part | Range of data for multiple parts |
| Command receives data element value | Command receives name and path of a delimited file containing the part data. |
| Less powerful, flexible | More powerful and flexible |

# Data Element Action

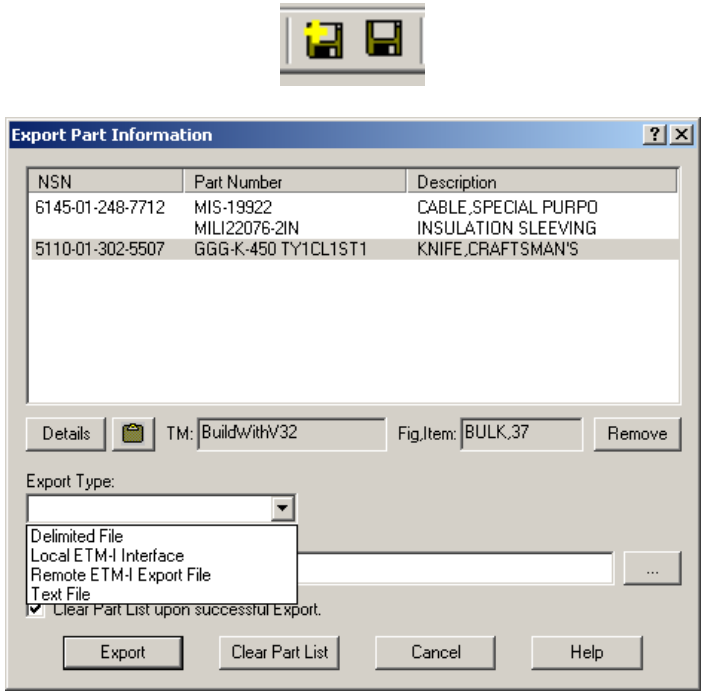
- Initiated when user double-clicks on a particular data element in the list.
- RPSTL simply create a process that runs the associated command.
- If no action is defined for selected element, it writes the element value to the clipboard.
- Commands can reference a parameter, or a relative path based on the destination data directory.
- An insertion marker is used to place the data element value within the command.




Button and Dialog to set data element action

# Export Function

- Build set of parts through ‘shopping cart’ style interface.
- Select from set of standard and custom export types (only standard shown).
- Data entry and produced output vary based on export type.
- Part data is formatted and transmitted according to export type.




RPSTL Reader buttons and dialog that implement the Export function.

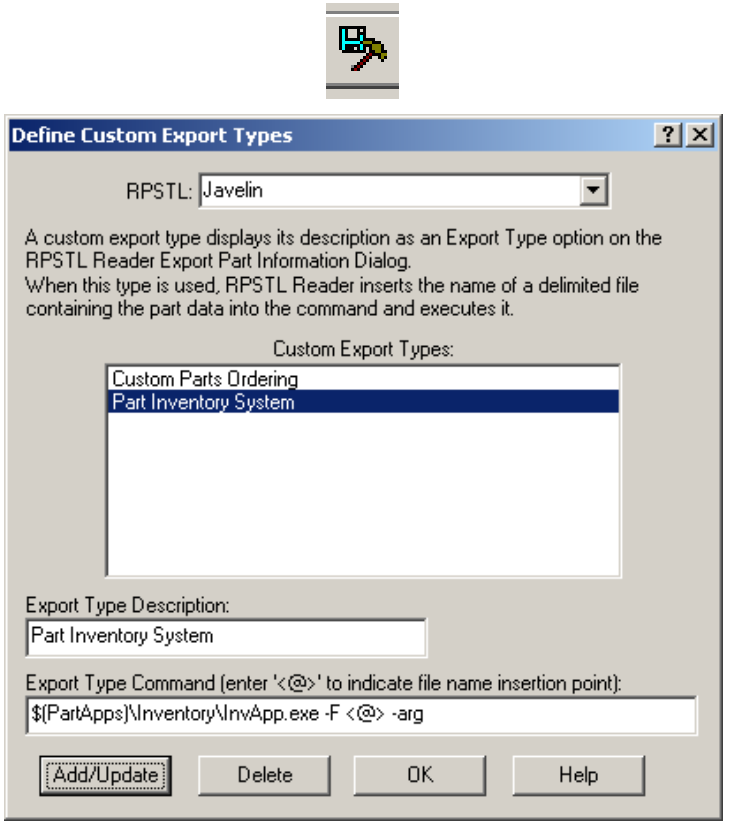
# Standard Export Types

Standard Export Types are available for all RPSTLs and have no authoring component.

| Export Type | Notes |
|---|---|
| Delimited File | User can select delimiter character. |
| Local ETM-I Interface | ETM-I is a program that manages repair shop workflow and provides a direct interface to ULLS-G. Our interface supports the parts ordering component. |
| Remote ETM-I Export File | |
| Text File | ASCII report similar in format to printed item report. |

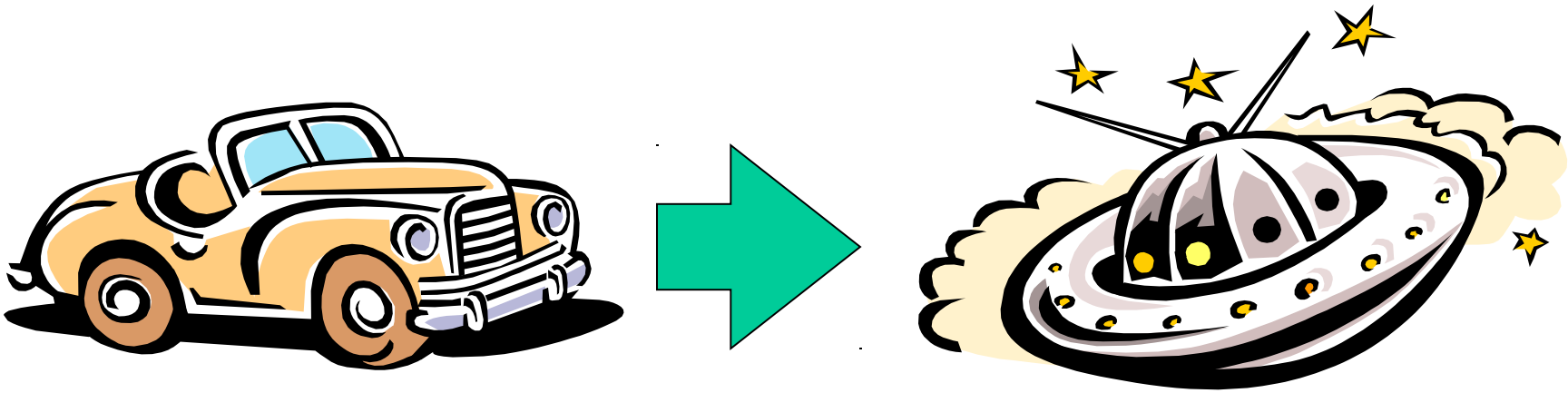
# Custom Export Types

- Defined through RPSTL Author.
- Apply only to the specific RPSTL.
- Appear after the standard export types in the Export dialog.
- Requires entry of a display name and a command.
- Passes part data through a delimited file whose name and location is passed to the command.
- Commands can reference a parameter, or a relative path based on the destination data directory.
- An insertion marker is used to place the name and path of the delimited file within the command.


Define Custom Export Types
RPSTL: Javelin
A custom export type displays its description as an Export Type option on the RPSTL Reader Export Part Information Dialog.
When this type is used, RPSTL Reader inserts the name of a delimited file containing the part data into the command and executes it.
Custom Export Types:
Custom Parts Ordering
Part Inventory System
Export Type Description:
Part Inventory System
Export Type Command (enter '<@>' to indicate file name insertion point):
$(PartApps)\Inventory\InvApp.exe -F <@> -arg
Add/Update
Delete
OK
Help

RPSTL Author button and dialog for creating custom export types.

# Looking Ahead

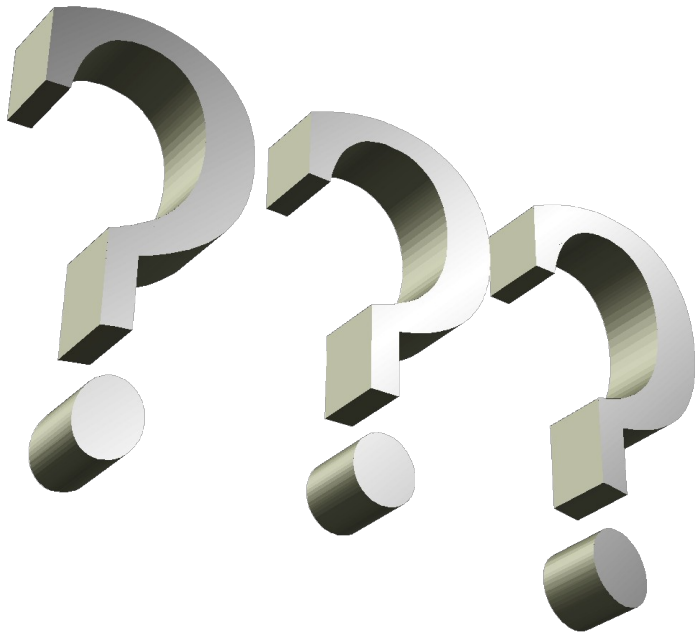


# Areas of Future Development

- Integration/Consolidation of IADS Reader programs.
- Initial Graphic and graphic navigation
- Consolidate Open and Search options.
- GUS and searching across RPSTLs
- Increasing Export and Element Action Flexibility
- Adding Import capabilities
- Expanded CGM tool
- Support for multiple document sets
- Tri-Service IETM Requirements
- Server-Based RPSTL processing for Internet access.

# What Else?

# Question and Answer



Don’t Forget Tomorrow’s Side Session!